

WHAT IS CLAIMED IS:

1. A method for enhancing source code for execution on a computer platform that has a capability to employ a memory file, said method comprising:
recognizing an occurrence of a first instruction in said source code that does not utilize said capability; and
supplementing said source code with a second instruction that utilizes said capability.
2. The method of claim 1, wherein said recognizing and said supplementing are performed when porting said source code from a first source file to a second source file.
3. The method of claim 1, wherein said supplementing provides said second instruction as part of a module for opening a memory file for use as a temporary work file during execution of said source code.
4. The method of claim 3, wherein said module is also for providing a handle for use by said source code to access said memory file subsequent to said opening of said memory file.
5. The method of claim 3,
wherein said first instruction is for opening a permanent file, and
wherein said module is also for reading data from said permanent file, and
writing said data to said memory file.
6. The method of claim 3,
wherein said first instruction is for reading data in EBCDIC format, and
wherein said module is also for reading said data, converting said data from EBCDIC format to ASCII format, and writing said data to said memory file in ASCII format.

7. The method of claim 3,
wherein said first instruction is for reading data in ASCII format, and
wherein said module is also for reading said data, and writing said data to
said memory file in ASCII format.

8. The method of claim 7, wherein said module is also for converting said
data from said memory file into EBCDIC format, and writing said data in
EBCDIC format to a permanent file.

9. A method for enhancing source code for execution on a computer
platform that has a capability to employ a memory file, comprising:
recognizing an occurrence of a first instruction in said source code that does
not utilize said capability; and
supplementing said source code with a module that opens a memory file for
use as a temporary work file during execution of said source code,
wherein said recognizing and supplementing are performed when porting
said source code from a first source file to a second source file.

10. A system for enhancing source code for execution on a computer
platform that has a capability to employ a memory file, said system comprising a
processor for:
recognizing an occurrence of a first instruction in said source code that does
not utilize said capability; and
supplementing said source code with a second instruction that utilizes said
capability.

11. The system of claim 10, wherein said processor performs said
recognizing and said supplementing when porting said source code from a first
source file to a second source file.

12. The system of claim 10, wherein said supplementing provides said second instruction as part of a module for opening a memory file for use as a temporary work file during execution of said source code.

13. The system of claim 12,
wherein said first instruction is for opening a permanent file, and
wherein said module is also for reading data from said permanent file, and
writing said data to said memory file.

14. The system of claim 12,
wherein said first instruction is for reading data in EBCDIC format, and
wherein said module is also for reading said data, converting said data from
EBCDIC format to ASCII format, and writing said data to said
memory file in ASCII format.

15. The system of claim 12,
wherein said first instruction is for reading data in ASCII format, and
wherein said module is also for reading said data, and writing said data to
said memory file in ASCII format.

16. A system for enhancing source code for execution on a computer platform that has a capability to employ a memory file, said system comprising a processor for:

recognizing an occurrence of a first instruction in said source code that does
not utilize said capability; and
supplementing said source code with a module that opens a memory file for
use as a temporary work file during execution of said source code,
wherein said recognizing and supplementing are performed when porting
said source code from a first source file to a second source file.

17. A storage media for enhancing source code for execution on a computer platform that has a capability to employ a memory file, said storage media comprising instructions for controlling a processor for:

recognizing an occurrence of a first instruction in said source code that does not utilize said capability; and
supplementing said source code with a second instruction that utilizes said capability.

18. The storage media of claim 17, wherein said instructions are for controlling said processor to perform said recognizing and said supplementing when porting said source code from a first source file to a second source file.

19. The storage media of claim 17, wherein said supplementing provides said second instruction as part of a module for opening a memory file for use as a temporary work file during execution of said source code.

20. The storage media of claim 19,
wherein said first instruction is for opening a permanent file, and
wherein said module is also for reading data from said permanent file, and
writing said data to said memory file.

21. The storage media of claim 19,
wherein said first instruction is for reading data in EBCDIC format, and
wherein said module is also for reading said data, converting said data from EBCDIC format to ASCII format, and writing said data to said memory file in ASCII format.

22. The storage media of claim 19,
wherein said first instruction is for reading data in ASCII format, and
wherein said module is also for reading said data, and writing said data to said memory file in ASCII format.

23. A storage media for enhancing source code for execution on a computer platform that has a capability to employ a memory file, said storage media comprising instructions for controlling a processor for:

recognizing an occurrence of a first instruction in said source code that does not utilize said capability; and

supplementing said source code with a module that opens a memory file for

use as a temporary work file during execution of said source code,

wherein said recognizing and supplementing are performed when porting

said source code from a first source file to a second source file.